



UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/821,537	03/28/2001	Randall K. Curey	P573C	6434

23586 7590 12/13/2006
ROBERT E MALM
16624 PEQUENO PLACE
PACIFIC PALISADES, CA 90272

EXAMINER

LEE, PHILIP C

ART UNIT	PAPER NUMBER
----------	--------------

2152

DATE MAILED: 12/13/2006

Please find below and/or attached an Office communication concerning this application or proceeding.



UNITED STATES PATENT AND TRADEMARK OFFICE

Commissioner for Patents
United States Patent and Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450
www.uspto.gov

MAILED

DEC 13 2006

Technology Center 2100

**BEFORE THE BOARD OF PATENT APPEALS
AND INTERFERENCES**

Application Number: 09/821,537

Filing Date: March 28, 2001

Appellant(s): CUREY ET AL.

Randall K. Curey
Daniel A. Tazartes
Kent T. Banno
John G. Mark
For Appellant

EXAMINER'S ANSWER

Art Unit: 2152

This is in response to the appeal brief filed 09/29/2006 appealing from the Office action mailed 10/22/2004.

(1) *Real Party in Interest*

A statement identifying by name the real party in interest is contained in the brief.

(2) *Related Appeals and Interferences*

The following are the related appeals, interferences, and judicial proceedings known to the examiner which may be related to, directly affect or be directly affected by or have a bearing on the Board's decision in the pending appeal:

Application 09/572,298, filed 05/16/2000, which was appealed to the Board of Patent Appeals and Interferences, then withdrawn from appeal by the patent office, and allowed as a patent US 6,829,763 on 12/07/2004.

(3) *Status of Claims*

The statement of the status of claims contained in the brief is correct.

(4) *Status of Amendments After Final*

The appellant's statement of the status of amendments after final rejection contained in the brief is correct.

(5) *Summary of Claimed Subject Matter*

The summary of claimed subject matter contained in the brief is correct.

(6) *Grounds of Rejection to be Reviewed on Appeal*

The appellant's statement of the grounds of rejection to be reviewed on appeal is correct.

(7) *Claims Appendix*

The copy of the appealed claims contained in the Appendix to the brief is correct.

(8) *Evidence Relied Upon*

4,109,311

Blum

8-1978

(9) *Grounds of Rejection*

The following ground(s) of rejection are applicable to the appealed claims:

1. Claims 1-49 are rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

A. The following words or phrases are not clearly understood and render the corresponding claims vague or indefinite :

a) "a plurality of software packages", claim 1, line 6, and claim 26, line 6, it is not clearly understood if "a plurality of software packages" is the same "the plurality of software packages" referred to in the same claims, lines 3-4, and line 5. If "a plurality of software packages" does refer to the same plurality, then the phrase should be "the plurality of software packages. However, if "a plurality of software packages does mean a subset of the previously defined "the plurality of software packages", or if it does mean a set of control software packages different from "the plurality of software packages", then it is not clearly understood on what bases, how or why, this subset of this set is being chosen/generated to be executed;

b) "each software package", claims 21 and 46, line 2; "a software package", claims 21 and 46, lines 3-4; and "another software package", claims 21 and 46, line 5, again as stated above, it is not clearly understood to which plurality of software packages this limitation is referring to.

c) "the plurality of software packages", claims 23 and 48, lines 1-2, again as stated above, it is not clearly understood to which plurality of software packages this limitation is referring to.

d) "a software package", claims 24 and 49, line. I , again as stated above, it is not clearly understood from which plurality of software packages this limitation is referring to.

e) "the presence of those software packages that are present", claims 22 and 47, lines 1-2, it is not clearly understood what "presence" is referring to, and where is the location of these software packages to detected., and software packages are identified by "those".

2. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

A person shall be entitled to a patent unless -

(b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

3. Claims 1, 8, 23, 25, 26, 33, and 48 are rejected under 35 U.S.C. 102(b) as being anticipated by Blum et al. (US 4,109,31 I), hereafter "Blum".

4. As to claim 1, Blum discloses the invention as claimed including a method for repetitively executing (col. 5, lines 35-42; and col. 6, lines 50-52) a plurality of software packages at one or more rates (the program requiring the greatest amount of processing time is allocated a greater number of time slices, col. 3, lines 47-54; fig. 3; col. 5, lines 24-42), utilizing a common set of computational resources (col. 3, lines 42-44), the method comprising: generating a sequence of time intervals for each of the plurality of software packages (col. 1, lines 11-23; col. 3, lines 47-54; and col. 6, lines 2-9), the time intervals belonging to one software package not overlapping the time intervals belonging to any other of the plurality of software packages (col. 3, lines 40-46); executing a plurality of software packages (abstract, lines

Art Unit: 2152

2-6; col. 2, lines 15-25), each software package being executed during the time intervals of its sequence of time intervals (abstract, lines 6-8; col. 1, lines 19-24; col. 6, lines 2-9; col. 7, lines 1-3).

5. As to claim 25, the claim is rejected for the same reasons as claim 1 above. In addition, Blum discloses an apparatus for practicing the method of claim 1 (Fig. 5).

6. As to claim 26, the claim is rejected for the same reasons as claim 1 above. In addition, Blum discloses an apparatus for repetitively executing (Fig. 5; and col. 5, lines 35-42; and col. 6, lines 50-52) a plurality of software packages at a plurality of rates (the program requiring the greatest amount of processing time is allocated a greater number of time slices, col. 3, lines 47-54; ; fig. 3; col. 5, lines 24-42), the apparatus comprising: a means for generating a sequence of time intervals for each of the plurality of software packages (col. 1, lines 11-23; col. 3, lines 47-54; and col. 6, lines 2-9), the time intervals belonging to one software package not overlapping the time intervals belonging to any other of the plurality of software packages (col. 3, lines 40-46); a means for executing a plurality of software packages (abstract, lines 2-6; col. 2, lines 15-25; and Fig. 5), each software package being executed, during the time intervals of its sequence of time intervals (abstract, lines 6-8; col. 1, lines 19-24; col. 6, lines 2-9; col. 7, lines 1-3).

7. As to claims 8 and 33, Blum discloses a software package is assigned its own dedicated memory region (fig. 3; col. 3, line 65 to col. 4, line 13; and col. 4, lines 50-54).

Art Unit: 2152

8. As to claims 23 and 48, Blum discloses a software package is independently compiled, linked, and loaded (col. 3, line 65 to col. 4, line 13; and col. 4, lines 50-54) (i.e., programs are transfer to control storage (loading) and executed (col. 4, lines 4-33) (compiling). Blum et al further teach pointer linking a program to be executed (col. 4, lines 47-61) (linking)).

9. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negated by the manner in which the invention was made.

10. Claims 10, 11, 13, 14, 18, 35, 36, 38, 39, and 43 are rejected under 35 U.S.C. 103(a) as being unpatentable over Blum et al. (US 4,109,311), hereafter "Blum" (i.e., in view of Official Notice).

11. As to claims 10, 11, 35, and 36, Blum does not necessarily disclose background tasks and foreground tasks. Within the software packages, however, it is well known in the art of task management, and would have been obvious to one skilled in the art at the time of the invention to divide the tasks in the software packages to foreground tasks and background tasks, where as the naming indicates, the foreground tasks have priority over the background tasks, a task such as an infinite loop may be running in the background.

Art Unit: 2152

12. As to claims 13, 14, 38, and 39, Blum does not necessarily disclose a failure log for logging failure of execution information. However, It is well known in the art of program execution, and would have been obvious to one skilled in the art at the time of the invention to log execution failures linked to software packages causing the failure in order to record different occurrences, reasons, and locations of execution failures for future modifications.

13. As to claims 18 and 43, Blum discloses software packages are assigned own memory partitions (col. 3, line 65 to col. 4, line 13; and col. 4, lines 50-54), but does not necessarily teach safety-critical and non-safety critical software. It would have been obvious to one skilled in the art at the time of the invention to place the safety-critical and the non-safety critical software in separate partitions (i.e., it would have been obvious to one skilled in the art at the time of the invention to place the safety-critical software and non- safety critical software in separate memory region in order to distinguish the different type of programs. Also, this would allow safety-critical software to be executed first before the non-safety critical software by placing the safety-critical software in the earliest time slices of the time cycle (memory regions)).

14. Claims 2-7, 9, 12, 15-17, 19-22, 24, 27-32, 34, 37, 40-42, 44-47, and 49 would be allowable if rewritten to overcome the rejection(s) under 35 U.S.C. 112, 2nd paragraph, set forth in this Office action and to include all of the limitations of the base claim and any intervening claims.

(10) Response to Argument

The examiner summarizes the various points raised by the appellant and addresses replies individually.

Appellant argued that:

- (1) whether claims 1, 25, and 26 are unpatentable under non-statutory double patenting in view of Mark et al., US 6,829,763.
- (2) whether claims 1 and 26 are unpatentable under 35 U.S.C. 112, second paragraph, as being indefinite.
- (3) whether claims 21 and 46 are unpatentable under 35 U.S.C. 112, second paragraph, as being indefinite.
- (4) whether claims 23 and 48 are unpatentable under 35 U.S.C. 112, second paragraph, as being indefinite.
- (5) whether claims 24 and 49 are unpatentable under 35 U.S.C. 112, second paragraph, as being indefinite.
- (6) whether claims 2 and 27 are unpatentable under 35 U.S.C. 112, second paragraph, as being indefinite.
- (7) “checksum test of a software package’s program memory” of claims 3 and 28 are not indefinite.
- (8) “and/or” of claims 6-7 and 31-32 are clear and not indefinite.
- (9) The terms “partition”, “safety-critical software”, and “non-safety-critical software” of claims 18 and 43 are clear and not indefinite.
- (10) “those software packages that are present” are clear and not indefinite.

(11) whether claim 25 is unpatentable under 35 U.S.C. 112, second paragraph, as being indefinite.

(12) whether claim 25 is unpatentable under 35 U.S.C. 101 for embracing or overlapping two different statutory classes.

(13) Blum et al fails to teach repetitively executing a plurality of software packages at one or more rates as claimed in claim 1.

(14) Blum et al fails to teach generating a sequence of time intervals for each of the plurality of software packages, the time intervals belonging to one software package not overlapping the time intervals belonging to any other of the plurality of software packages.

(15) Blum et al fails to teach executing a plurality of software packages, each software package being executed during the time intervals of its sequence of time intervals.

(16) Blum et al does not teach a program being assigned its own dedicated memory region.

(17) Blum et al does not teach programs are compiled, linked, and loaded.

(18) background/foreground execution of claims 10 and 35 are not common knowledge.

(19) Examiner does not assert the limitation of claim 11 and 36 as it is well known in the art.

(20) The limitation of claims 13 and 38 are not well known in the art.

(21) The limitation of linking a failure in execution to the software that caused the failure of claims 14 and 39 are not well known.

(22) Blum et al does not teach safety-critical or non-safety-critical software nor does Blum et al teaches partitions for storage of programs.

In reply to argument (1): a terminal disclaimer has been filed by the appellant on 02/24/2006. Accordingly, the non-statutory double patenting rejection has been withdrawn.

In reply to argument (2): on page 20 of the appeal brief, appellant states: "in the claim preambles a method (claim 1) and apparatus (claim 26) for repetitively executing "a plurality of software packages". This recitation provides the antecedent basis for "THE plurality of software packages" which appears in the first element of both claims. "A plurality of software packages" that appears in the second element of both claims, in accordance with common practice, denotes a different set of software packages. (It does not use the phrase "the plurality of software packages".) From the context (i.e. "each software package being executed during the time intervals of its sequence of time intervals"), we know that this different set of software packages is made up of one or more of the "plurality of software packages" referred to in the preamble and the first element." Appellant explains that "a plurality of software packages" that cited in the 2nd element (claim 1, line 6 and claim 26, line 6) is different from "a plurality of software packages" cited in the preamble, which denote different sets of software packages. Since the *same* terms "a plurality of software packages" cited in the preamble and 2nd element refer to different sets of software packages, they are causing ambiguity.

In reply to argument (3): on page 22 of the appeal brief, appellant states: “since claims 21 and 46 are limitations pertaining to the second elements of claims 1 and 26 (i.e. they deal with execution of the software packages), it is clear that they must be associated with the “plurality of software packages” that appears in the second element of both claims.” Examiner disagrees that “each package”, “a software package”, and “another software package” *must* be associated with the “plurality of software packages” recited in the second elements of claims 1 and 26. Appellant alleges that since claims 21 and 46 are limitations dealing with execution of the software packages, therefore, “each package”, “a software package”, and “another software package” are pertaining to the second element of claims 1 and 26. However, according to the same reasoning of appellant, “each package”, “a software package”, and “another software package” can also be pertained to “plurality of software packages” recited in the preamble of claims 1 and 26 because “plurality of software packages” recited in the preamble of claims 1 and 26 is also dealing with execution of software packages (i.e. repetitively executing of software packages). Since both “plurality of software packages” in the preamble and 2nd element of claims 1 and 26 have to do with execution, and “plurality of software packages” in the preamble and “plurality of software packages” in the 2nd element are different set of software packages (as discussed in reply to argument (2)), it is unclear which “plurality of software packages” of claims 1 and 26 is to be associated with “each package”, “a software package”, and “another software package” of claims 21 and 46.

In reply to argument (4): on page 22 of the appeal brief, appellant states: “claims 23 and 48 are obviously limitations of the second element of claims 1 and 26 respectively having to do with executing programs. It follows that the antecedent basis of “the plurality of software

Art Unit: 2152

packages” referred to in the claims is the phrase “a plurality of software packages” appearing in the second element of the claims.” For the same reason explained in reply to argument (3) above, “the plurality of software packages” of claims 23 and 48 can also be pertained to “plurality of software packages” recited in the preamble of claims 1 and 26 because “plurality of software packages” recited in the preamble of claims 1 and 26 is also dealing with execution of software packages (i.e. repetitively executing of software packages). It is still unclear which “plurality of software packages” of claims 1 and 26 is to be associated with “the plurality of software packages” of claims 23 and 48.

In reply to argument (5): Examiner disagrees with appellant for the same reason as in reply to argument (4).

In reply to argument (6): the 35 U.S.C. 112, second paragraph rejection of claims 2 and 27 has been withdrawn.

In reply to argument (7): on page 24 of the appeal brief, appellant states: “The software package has been checked for validity prior to being loaded into the software package's program memory in the computer system where it is to be executed. The question to be answered by the checksum test is whether the software package's program memory has successfully captured an error-free version of the software package and it is appropriate to refer to the test as being a “checksum test of a software package's program memory” rather than the clearly misleading description of the test as being a “checksum test of a software package”. It is indefinite to perform checksum process on a memory because it is known that checksum process should be performed on a software, not a memory. According to the same reasoning as appellant, the memory should also have been checked for validity prior to installing on a computer system. It

is unclear how a checksum process can be applied on a memory of a computer system.

Examiner suggests that the validity test should be expressed as a "checksum test of a software package loaded on a software package's program memory" rather than the clearly misleading description of the test as being a "checksum test of a software package's program memory" (i.e., just memory).

In reply to argument (8): the 35 U.S.C. 112, second paragraph rejection of claims 6-7 and 31-32 has been withdrawn.

In reply to argument (9): the 35 U.S.C. 112, second paragraph rejection of claims 18 and 43 has been withdrawn.

In reply to argument (10): on page 28 of the appeal brief, appellant states: "The term "those software packages that are present" is a newly-defined group of software packages (one or more of the "plurality of software packages" referred to in the second element of claims 1 and 26) and needs no antecedent basis." If "those software packages that are present" is a newly-defined group of software packages, then it is misleading to use the term "those". The term "those software packages that are present" can be considered as software packages mentioned previously. If appellant intends for the term "those software packages" to be referred to one or more of the "plurality of software packages" cited to in the second element of claims 1 and 26, then it has the same problem as addressed to in reply to argument (3), it is unclear which of "plurality of software packages" of claims 1 and 26 is referring to.

In reply to argument (11): the 35 U.S.C. 112, second paragraph rejection of claim 25 has been withdrawn.

In reply to argument (12): the 35 U.S.C. 101 of claim 25 has been withdrawn.

In reply to argument (13): on page 34 of the appeal brief, appellant states: "The rate at which a software package is repetitively executed is equal to the reciprocal of the repetition period. If the repetition period is not the same from one period to the next, one cannot speak of a software package having a "rate" of execution." On page 37, appellant further argue "assigning different numbers of time slices to different programs is not the same as repetitively executing different programs at the same or different rate." Examiner disagrees. Blum et al discloses execution of a plurality of programs by assigning index word or time slice to each of the plurality of program. In an example, four programs are being executed in a string of sixteen index words. Each index word addresses an instruction for one of the four programs. Each program can occupy anywhere from one-sixteenth to thirteen-sixteenths of the total computing time (col. 5, lines 5-42). This means for example, consider each of the four programs A, B, C, and D consisted of 4 instructions for each program, and the four programs are being executed in a string of eight index words, wherein eight index word consider to be one time cycle. If program A occupies 4 out of 8 index words, and programs B to D occupy the other 4 out of 8 index words, program A will be completely executed in one cycle (i.e., one rate). If program A occupies 3 out of 8 index words, and programs B to D occupy the other 5 out of 8 index words, program A will be completely executed in more than one cycle (i.e. another rate). After execution of the last index word in a cycle causes a return to the first index word to start another time slice cycle (col. 5, lines 35-37). This means that the programs are being repetitively executed. Accordingly, each of the four programs is being repetitively executed in one or more rates.

In reply to arguments (14) and (15): Blum et al teaches the operating time of the instruction execution unit is subdivided into a recurring set of time slice intervals (i.e., generating

Art Unit: 2152

a sequence of time intervals). The first time slice in the set is assigned to a first program, the second time slice is assigned to a second program, and so on. (the time intervals are belonging to one software package not overlapping the time intervals belonging to any other of the plurality of software packages) During any given time slice, one or two or a few instructions from the assigned program are executed. The instructions from the different programs are executed in an interleaved manner. (i.e. executing a plurality of software packages, each software package being executed during the time intervals of its sequence of time intervals) (col. 1, lines 11-23; col. 3, lines 35-54). On page 42 of the appeal brief, appellant further argue: "it is appropriate to invoke 35 U.S.C. 112, ¶ 6. See *In re Donaldson Co.*, 16 F.3d 1189, 29 USPQ2d 1845 (Fed. Cir. 1994). Limitation [3] qualifies in that Limitation [3] is simply a statement of a function to be performed without the recitation of acts for performing that function. Manual of Patent Examining Procedure, 8th Edition, May 2004 Revision, § 2181 I. Limitation [3] speaks only of executing a plurality of software packages, each software package being executed during the time intervals of its sequence of time intervals.

It follows from the words of Limitation [3] and further supported by Appellants' specification and drawings that the steps involved in performing the function of Limitation [3] are: (1) identify the next time interval, (2) identify the sequence of time intervals of which the next time interval is a member, (3) identify the program associated with the identified sequence of time intervals, and (4) execute the identified program beginning with the start of the next time interval. Appellants' Specification, p. 6, line 8 - p. 7, line 17.

There are no acts nor apparatus described in Blum et al. that are equivalent to those employed by appellants' in performing the function of Limitation [3]."

Examiner disagrees with appellant for invoking 35 U.S.C. 112, ¶ 6, the claims are interpreted in light of the specification, limitations from the specification are not read into the claims. See *In re Van Geuns*, 988 F.2d 1181, 26 USPQ2d 1057 (Fed. Cir. 1993).

In reply to argument (16): as shown in figure 3 of Blum et al, control storage 26 (i.e. memory) is assigned to different programs (PGM 0, 1, 2), each of the program is stored in the control storage in an assigned memory slot (i.e. memory region).

In reply to argument (17): Blum et al teaches programs are transfer to control storage (loading) and executed (col. 4, lines 4-33) (compiling). Blum et al further teach pointer linking a program to be executed (col. 4, lines 47-61) (linking).

In reply to argument (18): on page 45, last paragraph to page 6, 1st paragraph, appellant argue: “the examiner seems to ignore the complexities of the Blum et al. system...” The issue at hand is would it be obvious for one of ordinary skill in the art at the time of the invention to include the background and foreground tasks in executing a program, and there must be a reasonable expectation of success in combining background and foreground tasks in executing a program in Blum et al. One of ordinary skill in the art at the time of the invention would understand background and foreground tasks and the invention disclosed by Blum et al are implemented with software instruction executed on a processor. Hence, one of ordinary skill in the art at the time of the invention would have reasonable expectation of success in combine software codes of Blum et al’s invention with software codes of performing background/foreground tasks. On page 46, appellant provided an encyclopedia article as support for arguing the complexity of including background/foreground task. However, the encyclopedia article is not considered as evidence because the reference relied upon by the

appellant was not submitted pursuant to §§ 1.130, 1.131, or 1.132 in a timely matter. Reference to unentered evidence is not permitted in the brief. See § 41.63 for treatment of evidence submitted after appeal. Examiner acknowledges the quoting from the MPEP § 2144.03 on page 47 of the appeal brief. Examiner is willing to provide support of the “official notice” upon request from appellant.

In reply to argument (19): in paragraph 19 of the office action mailed on 10/22/04, an assertion that it is well known in the art that an infinite loop may be running in background task was made. Since claims 11 and 36 are dependent on claims 10 and 35 respectively, and from the grouping of rejection of claims 10, 11, 35, and 36, it is clear that it would have been obvious to combine the teaching of infinite loop for the same reason as cited for claims 10 and 35.

In reply to argument (20): On page 49, appellant rely upon expert opinion as support for arguing the limitation of logging in a logged file failure of a software execution is not well known. However, the expert opinion provided by the appellant is not considered as evidence because the reference relied upon by the appellant was not submitted pursuant to §§ 1.130, 1.131, or 1.132 in a timely matter. Reference to unentered evidence is not permitted in the brief. See § 41.63 for treatment of evidence submitted after appeal. Examiner acknowledges the quoting from the MPEP § 2144.03 on page 48 of the appeal brief. Examiner is willing to provide support of the “official notice” upon request from appellant.

In reply to argument (21): On pages 50-51, appellant rely upon computer science encyclopedia as support for arguing the limitation of linking a failure in execution to the software that caused the failure is not well known. However, the computer science encyclopedia provided by the appellant is not considered as evidence because the reference relied upon by the

Art Unit: 2152

appellant was not submitted pursuant to §§ 1.130, 1.131, or 1.132 in a timely matter. Reference to unentered evidence is not permitted in the brief. See § 41.63 for treatment of evidence submitted after appeal. Examiner acknowledges the quoting from the MPEP § 2144.03 on page 51 of the appeal brief. Examiner is willing to provide support of the “official notice” upon request from appellant.

In reply to argument (22): appellant’s argument of partition has been considered and addressed to in reply to argument (16). As stated in paragraph 21 of the office action mailed 10/22/2004, it would have been obvious to one skilled in the art at the time of the invention to place the safety-critical and the non-safety critical software in separate partitions. This means that it would have been obvious to one skilled in the art at the time of the invention to place the safety-critical software and non- safety critical software in separate memory region in order to distinguish the different type of programs. Also, this would allow safety-critical software to be executed first before the non-safety critical software by placing the safety-critical software in the earliest time slices of the time cycle (memory regions).

(11) *Related Proceeding(s) Appendix*

No decision rendered by a court or the Board is identified by the examiner in the Related Appeals and Interferences section of this examiner’s answer.

(12) *Conclusion*

For the above reasons, it is believed that the rejections should be sustained.

Art Unit: 2152

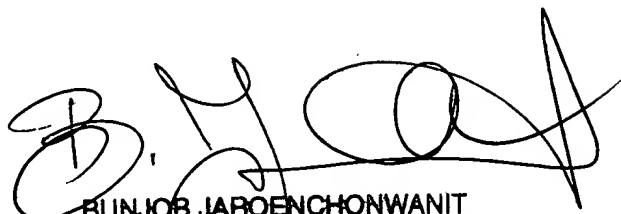
Respectfully submitted,

Philip Lee

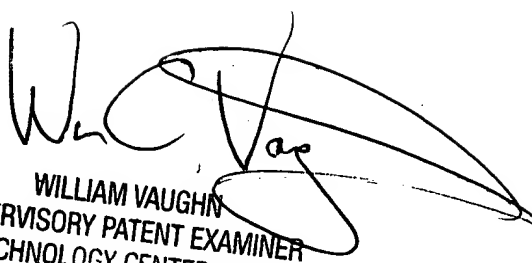
P.L.

December 07, 2006

Conferees:



BUNJOB JAROENCHONWANIT
SUPERVISORY PATENT EXAMINER



WILLIAM VAUGHN
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100